WHAT IS CLAIMED IS:

1. A CAN device that supports a plurality n of message objects, comprising:

a plurality of registers associated with each message object, including at least one object match ID register that contains a multi-bit object match ID field, and at least one object mask register that contains a multi-bit object mask field;

a CAN/CAL module that processes incoming messages, wherein the CAN/CAL module:

assembles a multi-bit screener ID from selected bits of each incoming message to be acceptance filtered;

compares the bits comprising the screener ID with corresponding bits of the object match ID field associated with each of at least designated ones of the plurality n of message objects, disregarding any bits of each object match ID field that are masked by corresponding bits of the associated object mask field; and,

determines whether any of the comparisons results in a match;

wherein any selected one or more bits of the object match ID field associated with each of the plurality n of message objects can be set to '1' or '0', and any selected one or more bits of the object mask field associated with each of the plurality n of message objects can be set to '1' or '0' in order to mask any selected one or more bits of the associated object match ID field, whereby the combination of the object match ID field and the object mask field associated with each of the plurality n of message objects comprises a fully programmable match and mask filter; and,

wherein $n \geq 3$.

2. The CAN device as set forth in Claim 1, wherein:

a received message to be acceptance filtered comprises a standard CAN frame; and,

the screener ID field comprises 11 bits of a CAN ID field of a header portion of the standard CAN frame, 8 bits of a first data byte of the standard CAN frame, 8 bits of a second data byte of the standard CAN frame, two don't care bits, and an IDE bit.

3. The CAN device as set forth in Claim 2, wherein the IDE bit is not maskable, and the two don't care bits are required to be masked.

4. The CAN device as set forth in Claim 1, wherein:

a received message to be acceptance filtered comprises an extended CAN frame; and,

the screener ID field comprises 29 bits of a CAN ID field of a header portion of the standard CAN frame, and an IDE bit.

5. The CAN device as set forth in Claim 4, wherein the IDE bit is not maskable.

6. The CAN device as set forth in Claim 1, wherein the CAN/CAL module has the capability to perform acceptance filtering on incoming messages comprising either standard or extended CAN frames.

7. The CAN device as set forth in Claim 1, wherein the plurality of registers further include a control register associated with each message object, wherein the control register associated with each message object contains an object enable field that is programmable for the purpose of enabling or disabling the associated message object.

8. The CAN device as set forth in Claim 7, wherein the control register associated with each message object further contains an object designation field that is programmable for the purpose of designating the associated message object as a receive or transmit message object, whereby the designated ones of the plurality n of message objects comprise all message objects that have been enabled and designated as a receive message object.

9. The CAN device as set forth in Claim 1, wherein:

the message objects are uniquely numbered; and,

if more than one match is detected, designating a lowest-numbered one of the message objects to be the matching message object.

10. The CAN device as set forth in Claim 1, wherein the plurality of registers further include an RTR register associated with each message object, wherein the RTR register associated with each message object contains an RTR handling enable field that is programmable for the purpose of enabling or disabling the associated message object to detect receipt of an incoming message that comprises an Remote Transmit Request frame and/or to transmit a Remote Transmit

Request frame.

11. The CAN device as set forth in Claim 1, wherein:

the CAN/CAL module has the capability of automatically assembling incoming messages that comprise multi-frame, fragmented messages;

the plurality of registers further include a fragmented message handling enable register associated with each message object, wherein the fragmented message handling enable register associated with each message object contains a fragmented message handling enable field that is programmable for the purpose of enabling or disabling the associated message object to receive multi-frame, fragmented messages for automatic assembly by the CAN/CAL module; and,

all of the designated ones of the plurality n of message objects are disabled to receive multi-frame, fragmented messages, and thus, are enabled only to receive single-frame, non-fragmented messages.

12. The CAN device as set forth in Claim 1, wherein the plurality of registers comprise memory-mapped registers.

13. The CAN device as set forth in Claim 1, further comprising a data memory space, wherein the plurality of memory-mapped registers are mapped to a respective portion of the data memory space.

14. The CAN device as set forth in Claim 1, further comprising a data memory space, wherein the plurality of memory-mapped registers are mapped to a respective dedicated RAM portion of the data memory space.

15. The CAN device as set forth in Claim 1, further comprising:

a data memory space;

a plurality of message buffers associated with respective ones of the message objects, the plurality of message buffers being located in the data memory space;

a DMA engine that enables the CAN/CAL module to directly access the message buffers.

16. The CAN device as set forth in Claim 15, wherein the plurality of memory-mapped registers are mapped to a respective portion of the data memory space.

- 32 -

17. The CAN device as set forth in Claim 15, wherein the plurality of memory-mapped registers are mapped to a respective dedicated RAM portion of the data memory space.

18. The CAN device as set forth in Claim 1, wherein the CAN device is a CAN microcontroller.

19. The CAN device as set forth in Claim 18, further comprising a processor core that runs CAN applications.

20. The CAN device as set forth in Claim 19, further comprising:
    a data memory space;
    a plurality of message buffers associated with respective ones of the message objects, the plurality of message buffers being located in the data memory space;
    a DMA engine that enables the CAN/CAL module to directly access the message buffers without interrupting the processor core.

21. The CAN device as set forth in Claim 20, wherein an incoming message for which a match is detected is stored by the DMA engine in the message buffer associated with the matching message object.

22. The CAN device as set forth in Claim 21, wherein:
    the message objects are uniquely numbered; and,
    if more than one match is detected, designating a lowest-numbered one of the message objects to be the matching message object.

23. The CAN device as set forth in Claim 15, wherein an incoming message for which a match is detected is stored by the DMA engine in the message buffer associated with the matching message object.

24. The CAN device as set forth in Claim 23, wherein:
    the message objects are uniquely numbered; and,
    if more than one match is detected, designating a lowest-numbered one of the message objects to be the matching message object.

- 33 -

25. A CAN microcontroller that supports a plurality n of message objects, comprising:

a processor core that runs CAN applications;

a data memory space;

a plurality of message buffers associated with respective ones of the message objects, the plurality of message buffers being located in the data memory space;

a plurality of registers associated with each message object, including at least one object match ID register that contains a multi-bit object match ID field, and at least one object mask register that contains a multi-bit object mask field;

a CAN/CAL module that processes incoming messages, wherein the CAN/CAL module:

assembles a multi-bit screener ID from selected bits of each incoming message to be acceptance filtered;

compares the bits comprising the screener ID with corresponding bits of the object match ID field associated with each of at least designated ones of the plurality n of message objects, disregarding any bits of each object match ID field that are masked by corresponding bits of the associated object mask field; and,

determines whether any of the comparisons results in a match;

a DMA engine that enables the CAN/CAL module to directly access the message buffers without interrupting the processor core;

wherein an incoming message for which a match is detected is stored by the DMA engine in the message buffer associated with the matching message object;

wherein any selected one or more bits of the object match ID field associated with each of the plurality n of message objects can be set to '1' or '0', and any selected one or more bits of the object mask field associated with each of the plurality n of message objects can be set to '1' or '0' in order to mask any selected one or more bits of the associated object match ID field, whereby the combination of the object match ID field and the object mask field associated with each of the plurality n of message objects comprises a fully programmable match and mask filter; and,

wherein $n \geq 3$.

26. The CAN microcontroller as set forth in Claim 25, wherein:

the message objects are uniquely numbered; and,

if more than one match is detected, designating a lowest-numbered one of the message

- 34 -

objects to be the matching message object.

27. The CAN microcontroller as set forth in Claim 25, wherein the plurality of registers comprise memory-mapped registers.

28. The CAN microcontroller as set forth in Claim 27, wherein the plurality of memory-mapped registers are mapped to a respective portion of the data memory space.

29. The CAN microcontroller as set forth in Claim 25, wherein:
a received message to be acceptance filtered comprises a standard CAN frame; and,
the screener ID field comprises 11 bits of a CAN ID field of a header portion of the standard CAN frame, 8 bits of a first data byte of the standard CAN frame, 8 bits of a second data byte of the standard CAN frame, two don't care bits, and an IDE bit.

30. The CAN microcontroller as set forth in Claim 29, wherein the IDE bit is not maskable, and the two don't care bits are required to be masked.

31. The CAN microcontroller as set forth in Claim 25, wherein:
a received message to be acceptance filtered comprises an extended CAN frame; and,
the screener ID field comprises 29 bits of a CAN ID field of a header portion of the standard CAN frame, and an IDE bit.

32. The CAN microcontroller as set forth in Claim 31, wherein the IDE bit is not maskable.

33. The CAN microcontroller as set forth in Claim 25, wherein the CAN/CAL module has the capability to perform acceptance filtering on incoming messages comprising either standard or extended CAN frames.

34. The CAN microcontroller as set forth in Claim 25, wherein the CAN/CAL module has the capability to perform acceptance filtering on incoming messages comprising Remote Transmit Request frames.

35. In a CAN device that supports a plurality n of message objects each of which has an

- 35 -

associated message buffer, at least one associated match ID register, and at least one associated mask register, a method for acceptance filtering incoming CAN frames, the method including the steps of:

programming the at least one match ID register associated with each of at least designated ones of the message objects by selectively setting each of the bits in a multi-bit match ID field contained therein to '1' or '0';

programming the at least one mask register associated with each of the at least designated ones of the message objects by selectively setting each of the bits in a multi-bit mask field contained therein to '1' or '0';

extracting a multi-bit screener ID field from a received CAN frame;

comparing the extracted screener ID field to the multi-bit match ID field stored in the at least one match ID register associated with each of the at least designated ones of the message objects, excluding from the comparison any bits of the match ID field masked by corresponding bits of the associated mask field stored in the at least one associated mask register; and,

if a match if found as a result of the comparing step, storing data bytes of the received CAN frame in the message buffer associated with the matching message object;

wherein $n \geq 3$.

36. The method as set forth in Claim 35, wherein the designated ones of the message objects comprise all enabled message objects that have been designated as receive message objects.

37. The method as set forth in Claim 35, wherein:

the message objects are uniquely numbered; and,

if more than one match is detected, designating a lowest-numbered one of the message objects to be the matching message object.

38. The method as set forth in Claim 35, wherein:

a received message to be acceptance filtered comprises a standard CAN frame; and,

the screener ID field comprises 11 bits of a CAN ID field of a header portion of the standard CAN frame, 8 bits of a first data byte of the standard CAN frame, 8 bits of a second data byte of the standard CAN frame, two don't care bits, and an IDE bit.

39. The method as set forth in Claim 38, wherein the IDE bit is not maskable, and the two don't

care bits are required to be masked.

40. The method as set forth in Claim 35, wherein:
     a received message to be acceptance filtered comprises an extended CAN frame; and,
     the screener ID field comprises 29 bits of a CAN ID field of a header portion of the standard CAN frame, and an IDE bit.

41. The method as set forth in Claim 40, wherein the IDE bit is not maskable.

42. The method as set forth in Claim 35, wherein:
     the CAN device is a CAN microcontroller that includes a processor core that runs CAN applications; and,
     the step of storing data bytes of the received CAN frame in the message buffer associated with the matching message object is performed without interrupting the processor core.